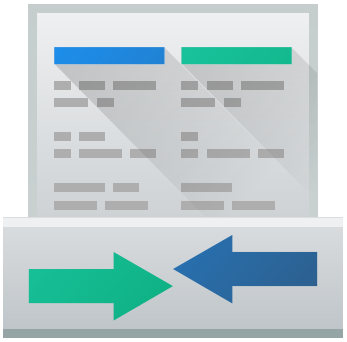
The KDiff3 Handbook



Contents

Abstract

KDiff3 is a file and folder diff and merge tool which

- compares and merges two or three text input files or folders,
- shows the differences line by line and character by character(!),
- provides an automatic merge-facility,
- has an editor for comfortable solving of merge-conflicts,
- provides network transparency via KIO,
- has options to highlight or hide changes in white-space or comments,
- supports Unicode, UTF-8 and other file encodings,
- prints differences,
- supports version control keyword and history merging.

This document describes KDiff3-version 1.10.

Chapter 1

Introduction

1.1 Yet Another Diff Frontend?

Several graphical diff tools exist. Why choose KDiff3? Let me say, why I wrote it.

KDiff3 started because I had to do a difficult merge. Merging is necessary when several people work on the same files in a project. A merge can be somewhat automated, when the merge-tool not only has the new modified files (called ″branches″), but also the original file (called ″base″). The merge tool will automatically choose any modification that was only done in one branch. When several contributors change the same lines, then the merge tool detects a conflict which must be solved manually.

The merge then was difficult because one contributor had changed many things and corrected the indentation in many places. Another contributor also had changed much text in the same file, which resulted in several merge conflicts.

The tool I used then, only showed the changed lines, but not what had changed within these lines. And there was no information about where only the indentation was changed. The merge was a little nightmare.

So this was the start. The first version could show differences within a line and showed white space differences. Later many other features were added to increase the usefulness.

For example if you want to compare some text quickly, then you can copy it to the clipboard and paste it into either diff window.

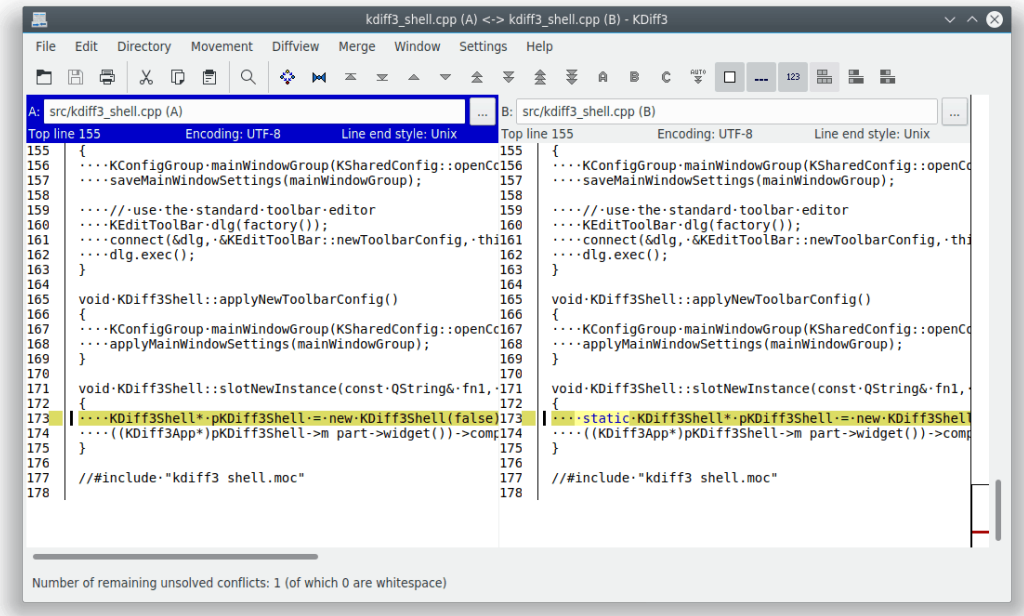
A feature that required a big effort was the folder comparison and merge facility, which turned the program almost into a full file browser.

I hope KDiff3 works for you too. Have fun!

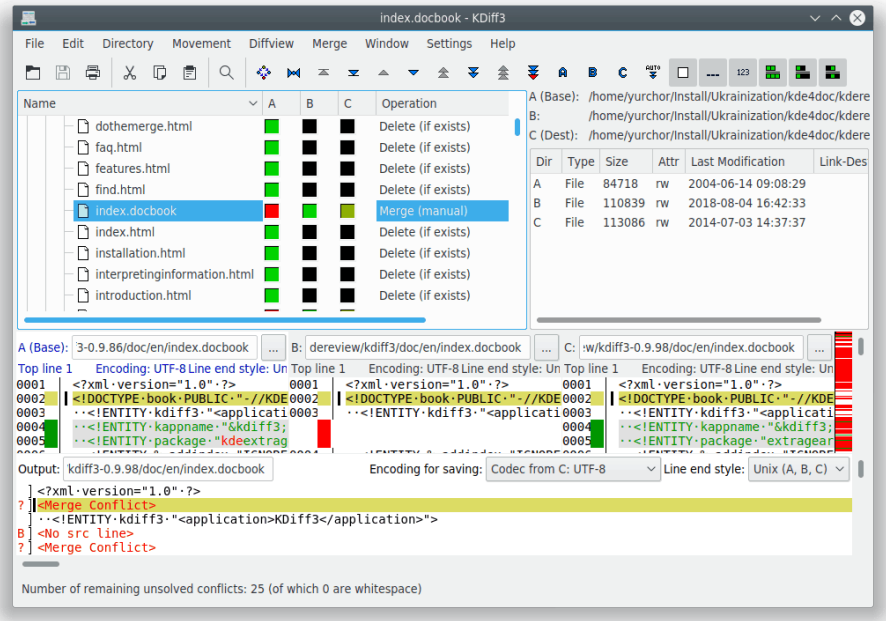
Joachim Eibl (2003)

1.2 Screenshots and Features

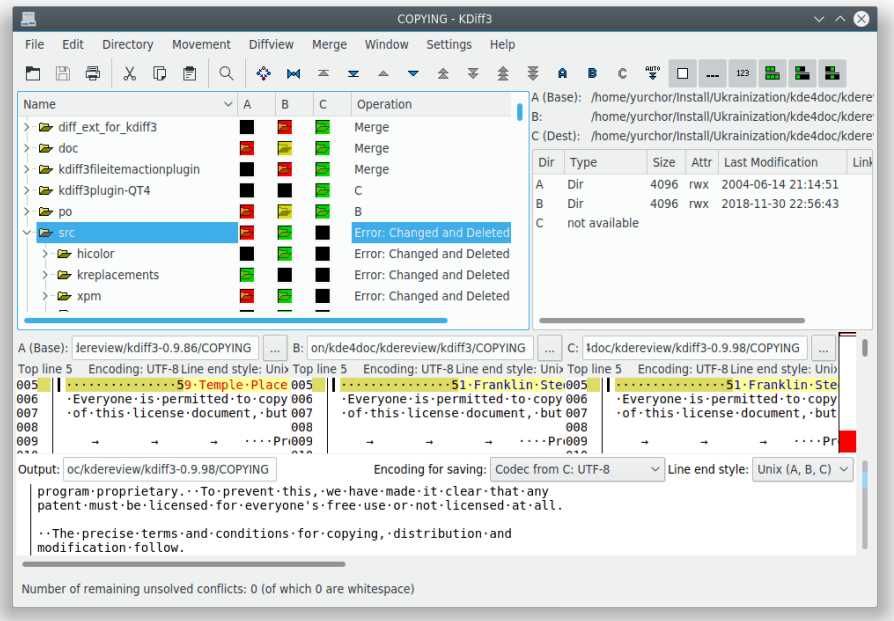
This screenshot shows the difference between two text files (using an early version of KDiff3):



3-way-merging is fully supported. This is useful if two people change code independently. The original file (the base) is used to help KDiff3 to automatically select the correct changes. The merge-editor below the diff-windows allows you to solve conflicts, while showing you the output you will get. You can even edit the output. This screenshot shows three input files being merged:



KDiff3 also helps you to compare and merge complete folders. This screenshot shows KDiff3 during a folder merge:



1.3 More Features

1.3.1 Line-By-Line And Char-By-Char Diff-Viewer

By using the possibilities of a graphical color display KDiff3 shows exactly what the difference is. When you have to do many code-reviews, you will like this.

```
0041 | <!---Date·and·version·information·of·the·documentat 0042 | <!---Date·and·version·information·of·the·documentat
0042 | Don't·forget·to·include·this·last·date·and·this·las 0043 | Don't·forget·to·include·this·last·date·and·this·las
0043 | need·them·for·translation·coordination·!           0044 | need·them·for·translation·coordination·!
0044 | Please·respect·the·format·of·the·date·(YYYY-MM-DD)· 0045 | Please·respect·the·format·of·the·date·(YYYY-MM-DD)·
0045 | (V.MM.LL),·it·could·be·used·by·automation·scripts. 0046 | (V.MM.LL),·it·could·be·used·by·automation·scripts.
0046 | Do·NOT·change·these·in·the·translation.···>         0047 | Do·NOT·change·these·in·the·translation.···>
0047 |                                                    0048 |
0048 | <date>2004-05-29</date>                            0049 | <date>2018-04-30</date>
0049 | <releaseinfo>0.9.84</releaseinfo>                  0050 | <releaseinfo>1.07.00</releaseinfo>
```

1.3.2 See White-Space Differences At One Glance

Spaces and tabs that differ appear visibly. When lines differ only in the amount of white space this can be seen at one look in the summary column on the left side. (No more worries when people change the indentation.)

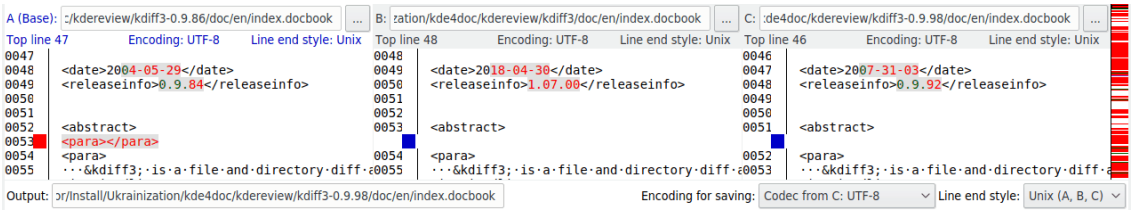
```
Start·from·commandline:                    083 | ···Start·from·commandline:
··Comparing·2·files:·····kdiff3·file1·file2 084 | ···--Comparing·2·files:·····kdiff3·file1·fi
··Merging·2·files:·······kdiff3·file1·file2 085 | ···--Merging·2·files:·······kdiff3·file1·fi
··Comparing·3·files:·····kdiff3·file1·file2 086 | ···--Comparing·3·files:·····kdiff3·file1·fi
··Merging·3·files:·······kdiff3·file1·file2 087 | ···--Merging·3·files:·······kdiff3·file1·fi
·····Note·that·file1·will·be·treated·as·bas 088 | ·····Note·that·file1·will·be·treated·as·
```

1.3.3 Triple-Diff

Analyze three files and see where they differ.

The left/middle/right windows are named **A**/**B**/**C** and have the blue/green/magenta color respectively.

If one file is the same and one file is different on a line then the color shows which file is different. The red color means that both other files are different.



1.3.4 Comfortable Merge Of Two Or Three Input Files

KDiff3 can be used to merge two or three input files and automatically merges as much as possible. The result is presented in an editable window where most conflicts can be solved with a single mouseclick: Select the respective **...** buttons at the **A**/ **B**/**C** windows to select the source that should be used. You can also select more than one source. Since this output window is an editor even conflicts which need further corrections can be solved here without requiring another tool.

1.3.5 And ...

- Fast navigation via buttons.
- A mouse-click into a summary column sync's all windows to show the same position.
- Select and copy from any window and paste into the merge result window.
- Overview column that shows where the changes and conflicts are.
- The colors are adjustable to your specific preferences.
- Adjustable Tab size.
- Option to insert spaces instead of tabs.
- Open files comfortably via dialog or specify files on the command line.
- Search for strings in all text windows with **Edit** → **Find** (**Ctrl-F**) and **Edit** → **Find Next** (**F3**) menu items.
- Show the line numbers for each line.
- Paste clipboard or drag text into a diff input window.
- Network transparency via KIO.
- Can be used as diff-viewer from the KDevelop 3.
- Word-wrap for long lines.
- Support for Unicode, UTF-8 and other codecs.
- Support for right to left languages.
- ...

Chapter 2

File Comparison And Merge

2.1 Command-Line Options

2.1.1 Comparing 2 files:

```
kdiff3 file1 file2
```

2.1.2 Merging 2 files:

```
kdiff3 file1 file2 -m
kdiff3 file1 file2 -o outputfile
```

2.1.3 Comparing 3 files:

```
kdiff3 file1 file2 file3
```

2.1.4 Merging 3 files:

```
kdiff3 file1 file2 file3 -m
kdiff3 file1 file2 file3 -o outputfile
```

Note that *file1* will be treated as base of *file2* and *file3*.

2.1.5 Special case: Files with the same name

If all files have the same name but are in different folders, you can reduce typework by specifying the filename only for the first file, e.g.:

```
kdiff3 folder1/filename folder2 folder3
```

2.1.6 Commandline for starting a folder comparison or merge:

This is very similar, but now it's about folders.

```
kdiff3 folder1 folder2
kdiff3 folder1 folder2 -o destdir
kdiff3 folder1 folder2 folder3
kdiff3 folder1 folder2 folder3 -o destdir
```

For folder comparison and merge you can continue to read here.

2.1.7 Other command line options

To see all available command line options type

```
kdiff3 --help
```

Example output:

```
Options:
  -m, --merge               Merge the input.
  -b, --base file           Explicit base file. For compatibility with  ↩
      certain tools.
  -o, --output file         Output file. Implies -m. E.g.: -o newfile.txt
  --out file                Output file, again. (For compatibility with  ↩
      certain tools.)
  --noauto                  Ignore --auto and always show GUI.
  --auto                    No GUI if all conflicts are auto-solvable. ( ↩
      Needs -o file)
  --L1 alias1               Visible name replacement for input file 1 (base ↩
      ).
  --L2 alias2               Visible name replacement for input file 2.
  --L3 alias3               Visible name replacement for input file 3.
  -L, --fname alias         Alternative visible name replacement. Supply  ↩
      this once for every input.
  --cs string               Override a config setting. Use once for every  ↩
      setting. E.g.: --cs "AutoAdvance=1"
  --confighelp              Show list of config settings and current values ↩
      .
  --config file             Use a different config file.
```

The option `--cs` allows you to adjust a configuration value that is otherwise only adjustable via the configure dialogs. But be aware that when KDiff3 then terminates the changed value will be stored along with the other settings. With `--confighelp` you can find out the names of the available items and current values.

Via `--config` you can specify a different config file. When you often use KDiff3 with completely different setups this allows you to easily switch between them.

2.1.8 Ignorable command line options

Many people want to use KDiff3 with some version control system. But when that version control system calls KDiff3 using command line parameters that KDiff3 doesn't recognise, then KDiff3 terminates with an error. The **Integration** item in the settings dialog allow to specify command line parameters that should be ignored by KDiff3. They will appear in the usage help like in this example:

```
--foo                    Ignored. (User defined.)
```

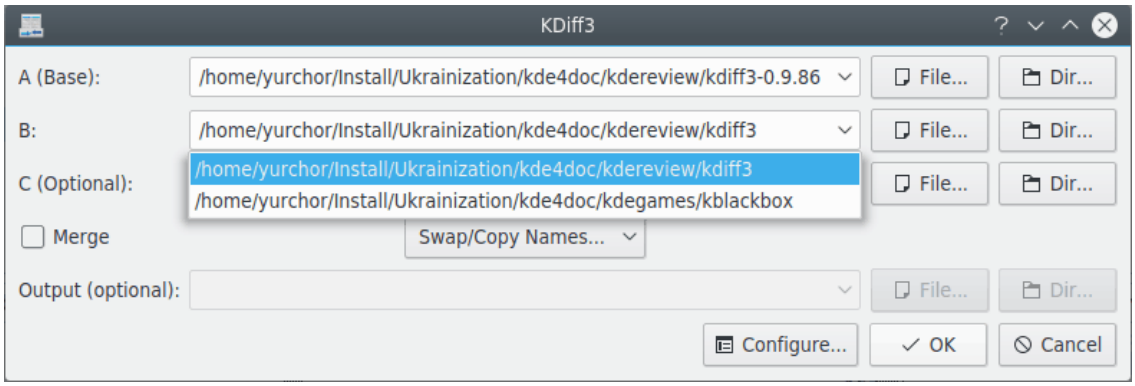
Command line options to ignore:

A list of options, separated via semicolon ';'. When one of these options appears on the commandline, then KDiff3 will ignore it and run without reporting an error. (Default is "u;query;html;abort").

When this isn't enough, then it is recommended to write a shell script that does the option translation.

2.2 Open-Dialog

Since many input files must be selectable, the program has a special open dialog:



The open dialog allows you to edit the filenames by hand, selecting a file via the file-browser via the **File...** button or allows you to choose recent files from the drop-down lists. If you open the dialog again, then the current names still remain there. The third input file is not required. If the entry for **C** remains empty, then only a two file diff analysis will be done.

You can also select a folder via the **Folder...** button. If for **A** a folder is specified then a folder-comparison /merge starts. If **A** specifies a file but **B**, **C** or the output specify a folder, then KDiff3 uses the filename from **A** in the specified folders.

If **Merge** check box is selected, then the **Output** line becomes editable. But it is not required to specify the output filename immediately. You can also postpone this until saving.

The **Configure...** button opens the settings dialog, so that you can set the options before running the analysis.

2.3 Paste and Drop Input

Sometimes you want to compare parts of a text that is not an own file. KDiff3 also allows you to paste text from the clipboard into the diff input window that has the focus. The diff analysis happens immediately then. In the open dialog you need not specify files then, but just close it via **Cancel** button.

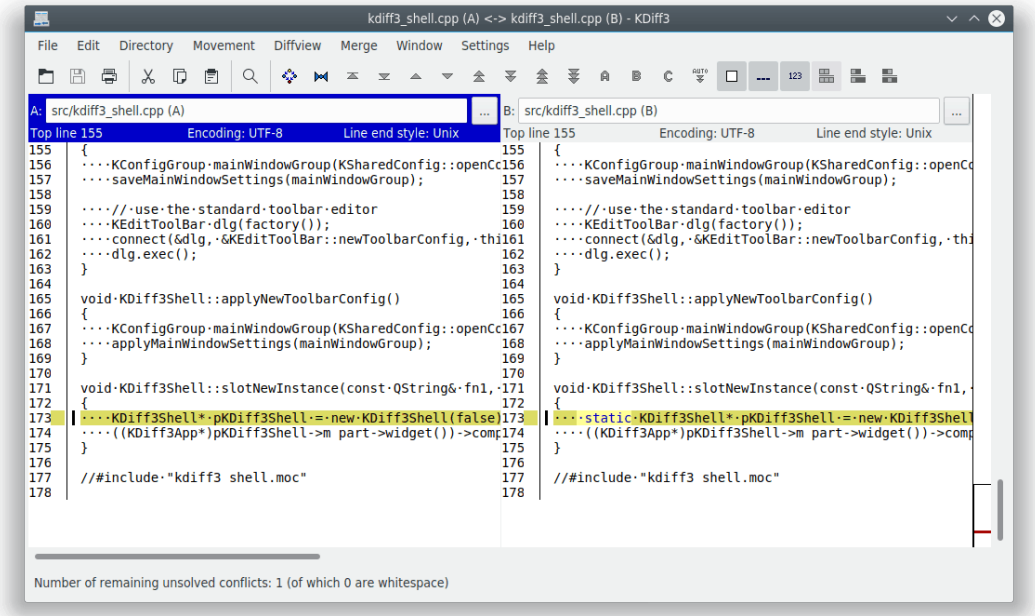
You can also use drag and drop: Drag a file from a file manager or selected text from an editor and drop it onto a diff input window.

What's the idea? Sometimes a file contains two similar functions, but checking how similar they really are is a big effort if you first must create two files and then load them. Now you can simply copy, paste and compare the relevant sections.

> **NOTE**
> Currently you cannot drag anything from KDiff3. Only dropping in the diff input is supported.

> **WARNING**
> Some editors still interpret the drag and drop into another program like cut (instead of copy) and paste. Your original data might be lost then.

2.4 Comparing Files And Interpreting The Information In The Input Windows



2.4.1 Info Line

At the top of each text window is its "info line". The info lines of the input windows contain a letter **A**, **B** or **C**, the editable filename, a button for browsing, and the line number of the first visible line in the window. (Note that window **C** is optional.) Each info line appears in a different color.

When you selected another file via browsing or finished editing the filename here by pressing **Enter**, the new file will be loaded and compared with the already loaded file(s).

2.4.2 Coloring

The three input windows are assigned the letters **A**, **B** and **C**. **A** has color blue, **B** has green and **C** has magenta. (These are the defaults, but can be changed in the **Color** item in the settings dialog.)

When a difference is detected then the color shows which input file differs. When both other input files differ then the color used to express this is red by default (**Conflict color** option in the **Color** item in the settings dialog). This colorscheme is especially useful in the case of three input files, which will be seen in the next section (Merging).

2.4.3 Summary Column

Left of each text is the ″summary″ column. If differences occurred on a line then the summary column shows the respective color. For a white-space-only difference the summary is chequered. For programming languages where white space is not so important this is useful to see at one glance if anything of importance was modified. (In C/C++ white space is only interesting within strings, comments, for the preprocessor, and some only very esoteric situations.)

The vertical line separating the summary column and the text is interrupted if the input file had no lines there. When word-wrap is enabled then this vertical line appears dotted for wrapped lines.

2.4.4 Overview Column

On the right side a ″overview″ column is visible left of the vertical scrollbar. It shows the compressed summary column of input **A**. All the differences and conflicts are visible at one glance. When only two input windows are used, then all differences appear red here because every difference is also a conflict. A black rectangle frames the visible part of the inputs. For very long input files, when the number of input lines is bigger than the height of the overview column in pixels, then several input lines share one overview line. A conflict then has top priority over simple differences, which have priority over no change, so that no difference or conflict is lost here. By clicking into this overview column the corresponding text will be shown.

2.4.5 Manually Aligning Lines

Sometimes the algorithm places the wrong lines next to each other. Or you want to compare one piece of text with text at a completely different position in the other file. For these situations you can manually instruct KDiff3 to align certain lines. Mark the text for which you want to improve the alignment with the mouse as you would for copy and paste in the first diff view and then choose **Diffview** → **Add Manual Diff Alignment** menu item (**Ctrl-Y**). An orange bar will appear in the summary column next to the chosen text. Repeat this for the second and (if available) third diff view. KDiff3 will immediately recalculate the differences everytime you do this, and will align the chosen lines. Of course some of the previously matching lines in between might not match anymore.

Currently merging doesn't support the use of manual diff help.

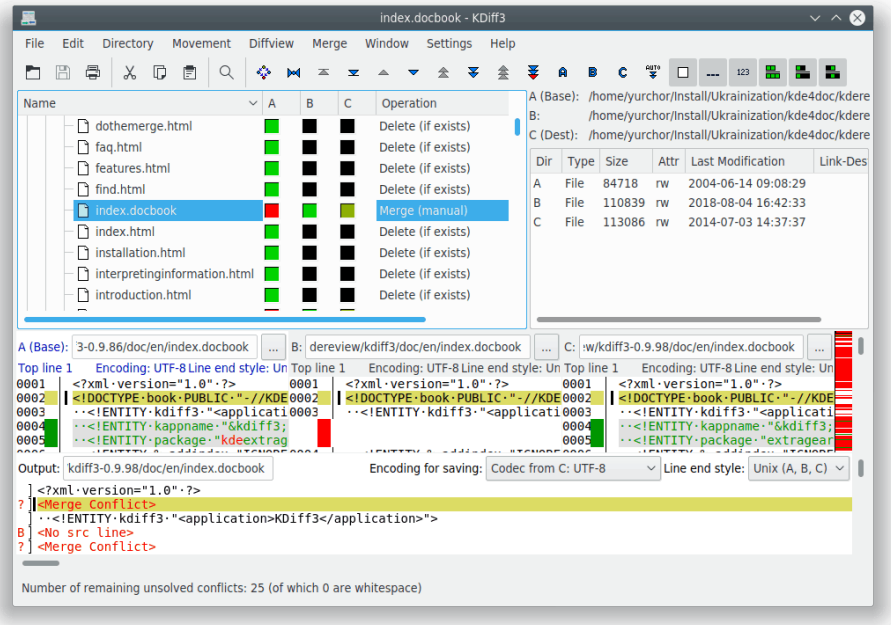
2.4.6 Manually Joining and Splitting Diff Sections

In some cases KDiff3 will see too many or too few diff sections for merging. In such a case you can join or split existing sections.

Add new sections by first selecting text in the lines that belong together in either input window (as for copying to the clipboard). Then choose **Merge** → **Split Diff At Selection** menu item. Splits will be added above the first line and below the last line of the selected text. If you only want to add one section, then select text beginning at another section-split.

For joining sections in either input window select something in the lines from the sections to join. (You can join several sections in one step too.) Then choose **Merge** → **Join Selected Diffs** menu item.

2.5 Merging And The Merge Output Editor Window



The merge output editor window (below the diff input windows) also has an info line at the top showing **Output:**, the filename and *[Modified]* tag if you edited something. Usually it will contain some text through the automatic merge facilities, but often it will also contain conflicts.

!!! Saving is disabled until all conflicts are resolved !!! (Use the **Go to Previous/Next Unsolved Conflict** buttons to find the remaining conflicts.)

With only two input files every difference is also a conflict that must be solved manually.

With three input files the first file is treated as base, while the second and third input files contain modifications. When at any line only either input **B** or input **C** have changed but not both then the changed source will automatically be selected. Only when **B** and **C** have changed on the same lines, then the tool detects a conflict that must be solved manually. When **B** and **C** are the same, but not the same as **A**, then **C** is selected.

2.5.1 The Summary Column

The merge output editor window also has a summary column on the left. It shows the letter of the input from which a line was selected or nothing if all three sources where equal on a line. For conflicts it shows a questionmark "?" and the line shows "<Merge Conflict>", all in red. Because solving conflicts line by line would take very long, the lines are grouped into groups that have the same difference and conflict characteristics. But only-white-space-conflicts are separated from non-white-space-conflicts in order to ease the merging of files were the indentation changed for many lines.

2.5.2 Setting The Current Group And Synchronising Merge And Diff View Position

When clicking into the summary column with the left mouse button in either window then the beginning of the group belonging to that line will shown in all windows. This group then becomes the "current group". It is highlighted with the **Current range diff background color** option of **Integration** item in the settings dialog and a black bar appears on the left side of the text.

2.5.3 Choosing Inputs A, B or C For Current Conflict And Editing

The Button bar below the menubar contains three input selector buttons containing the letters **A**, **B** and **C**. Click the input selector button to insert (or remove if already inserted) the lines from the respective source. To choose the lines from several inputs click the respective buttons in the needed order. For example if you want that the lines from **B** appear before the lines from **A** in the output, first click **B**, then **A**.

If you use the **Automatically Go to Next Unsolved Conflict After Source Selection** button (see Auto-Advance section), you should disable this before choosing lines from several inputs or if you want to edit the lines after your choice. Otherwise KDiff3 will jump to the next conflict after choosing the first input.

It is often helpful directly edit the merge output. The summary column will show ″m″ for every line that was manually modified. When for instance the differences are aligned in a way that simply choosing the inputs won't be satisfactory, then you can mark the needed text and use normal copy and paste to put it into the merge output.

Sometimes, when a line is removed either by automatic merge or by editing and no other lines remain in that group, then the text <No src line> will appear in that line. This is just a placeholder for the group for when you might change your mind and select some source again. This text won't appear in the saved file or in any selections you want to copy and paste.

The text ″<Merge Conflict>″ will appear in the clipboard if you copy and paste some text containing such a line. But still be careful to do so.

2.5.4 Choosing Input A, B, or C for All Conflicts

The normal merge will start by solving simple conflicts automatically. But the **Merge** menu provides some actions for other common needs. If you have to select the same source for most conflicts, then you can choose **A**, **B** or **C** everywhere, or only for the remaining unsolved conflicts, or for unsolved white space conflicts. If you want to decide every single delta yourself, you can **Set Deltas to Conflicts**. Or if you want to return to the automatic choices of KDiff3 then select **Automatically Solve Simple Conflicts**. KDiff3 then restarts the merge. For actions that change your previous modifications KDiff3 will ask for your confirmation before proceeding.

> NOTE
> When choosing either source for unsolved white space conflicts and the **Ignore numbers (treat as white space)** or **Ignore C/C++ comments (treat as white space)** options at **Diff** item in the settings dialog are used then changes in numbers or comments will be treated like white space too.

2.5.5 Automatic Merge of Version Control Keywords and History (Log)

Many version control systems support special keywords in the file. (e.g. ″$Date$″, ″$Header$″, ″$Author$″, ″Log″, etc.) During the check-in the version control system (VCS) changes these lines. For instance ″$Date$″ will turn into ″$Date: 2005/03/22 18:45:01 $″. Since this line will be different in every version of the file, it would require manual interaction during the merge.

KDiff3 offers automatic merge for these items at **Merge** item in the settings dialog. For simple lines that match the **Auto merge regular expression** option in all input-files KDiff3 will choose the line from **B** or - if available - from **C**. (Additionally it is necessary that the lines in question line up in the comparison and the previous line contains no conflict.) This auto merge can either be run immediately after a merge starts (activate the option **Run regular expression auto merge on merge start**) or later via **Merge** → **Run Regular Expression Auto Merge** menu item.

Automatic merge for version control history (also called ″log″) is also supported. The history automerge can either run immediately when the merge starts by activating the option **Merge**

version control history on merge start at **Merge** item in the settings dialog or later via the **Merge** → **Automatically Solve History Conflicts** menu item.

Usually the version control history begins with a line containing the keyword "Log". This must be matched by the **History start regular expression:** option. KDiff3 detects which subsequent lines are in the history by analysing the leading characters that came before the "Log" keyword. If the same "leading comment" characters also appears in the following lines, then they are also included in the history.

During each check-in the VCS writes a unique line specifying version-, date- and time-information followed by lines with user comments. These lines form one history-entry. This history section grows with every check-in and the most recent entries appear at the top (after the history start line).

When for parallel development two or more developers check-in a branch of the file then the merge history will contain several entries that appear as conflicts during the merge of the branches. Since merging these can become very tedious, KDiff3 offers support with two possible strategies: Just insert the history information from both contributors at the top or sort the history information by a user defined key.

The just-insert-all-entries-method is easier to configure. KDiff3 just needs a method to detect, which lines belong to one history entry. Most VCS insert an empty line after each history entry. If there are no other empty lines, this is a sufficient criterion for KDiff3. Just set an empty **History entry start regular expression** at **Merge** item in the settings dialog. If the empty line criterion isn't sufficient, you can specify a regular expression to detect the history entry start.

Note that KDiff3 will remove duplicate history entries. If a history entry appeared several times in the history of a input file, only one entry will remain in the output.

If you want to sort the history, then you have to specify how the sort key should be built. Use parentheses in the **History entry start regular expression** at **Merge** item in the settings dialog to group parts of the regular expression that should later be used for the sort key. Then specify the **History entry start sort key order** option specifying a comma "," separated list of numbers referring to the position of the group in the regular expression.

Because this is not so easy to get right immediately, you are able to test and improve the regular expressions and key-generation in a dedicated dialog by pressing the **Test your regular expressions** button.

Example: Assume a history that looks like this:

```
/*****************************************************************************
** HISTORY:    $Log: \toms_merge_main_view\MyApplication\src\ ↩
   complexalgorithm.cpp $
**
**     \main\integration_branch_12   2 Apr 2001 10:45:41   tom
**  Merged branch simon_branch_15.
**
**     \main\henry_bugfix_branch_7\1   30 Mar 2001 19:22:05   henry
**  Improved the speed for subroutine convertToMesh().
**  Fixed crash.
*****************************************************************************/
```

The history start line matches the regular expression ".*\$Log.*\$.*". Then follow the history entries.

The line with the "Log" keyword begins with two "*" after which follows a space. KDiff3 uses the first non-white-space string as "leading comment" and assumes that the history ends in the first line without this leading comment. In this example the last line ends with a string that also starts with two "*", but instead of a space character more "*" follow. Hence this line ends the history.

If history sorting isn't required then the history entry start line regular expression could look like this. (This line is split in two because it wouldn't fit otherwise.)

```
\s*\\main\\\S+\s+[0-9]+ (Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec)
 [0-9][0-9][0-9][0-9] [0-9][0-9]:[0-9][0-9]:[0-9][0-9]\s+.*
```

For details about regular expressions please see the regular expression documentation. Note that "\s" (with lowercase "s") matches any white space and "\S" (with uppercase "S") matches any non-white-space. In our example the history entry start contains first the version info with reg. exp. "\\main\\\S+", the date consisting of day "[0-9]+", month "(Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec)" and year "[0-9][0-9][0-9][0-9]", the time "[0-9][0-9]:[0-9][0-9]:[0-9][0-9]" and finally the developers login name ".*".

Note that the "leading comment" characters (in the example "**") will already be removed by KDiff3 before trying to match, hence the regular expression begins with a match for none or more white-space characters "\s*". Because comment characters can differ in each file (e.g. C/C++ uses other comment characters than a Perl script) KDiff3 takes care of the leading comment characters and you should not specify them in the regular expression.

If you require a sorted history. Then the sortkey must be calculated. For this the relevant parts in the regular expression must be grouped by parentheses. (The extra parentheses can also stay in if history sorting is disabled.)

```
\s*\\main\\(\S+)\s+([0-9]+) (Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov| ↩
    Dec)
 ([0-9][0-9][0-9][0-9]) ([0-9][0-9]:[0-9][0-9]:[0-9][0-9])\s+(.*)
```

The parentheses now contain 1. version info, 2. day, 3. month, 4. year, 5. time, 6. name. But if we want to sort by date and time, we need to construct a key with the elements in a different order of appearance: First the year, followed by month, day, time, version info and name. Hence the sortkey order to specify is "4,3,2,5,1,6".

Because month names aren't good for sorting ("Apr" would be first) KDiff3 detects in which order the month names were given and uses that number instead ("Apr" -> "04"). And if a pure number is found it will be transformed to a 4-digit value with leading zeros for sorting. Finally the resulting sort key for the first history entry start line will be:

```
2001 04 0002 10:45:41 integration_branch_12   tom
```

For more information also see Merge Settings section.

2.6 Navigation And Editing

Much navigation will be done with the scroll bars and the mouse but you can also navigate with the keys. If you click into either window then you can use the shortcuts **Left**, **Right**, **Up**, **Down**, **PgUp**, **PgDn**, **Home**, **End**, **Ctrl-Home** and **Ctrl-End** as you would in other programs. The overview-column next to the vertical scroll bar of the input files can also be used for navigating by clicking into it.

You can also use the wheel mouse to scroll up and down.

In the merge output editor you can also use the other keys for editing. You can toggle between insert and overwrite mode with the **Ins** key. (Default is insert-mode.)

A click with the left mouse button into any summary column will synchronise all windows to show the beginning of the same group of lines (as explained in section Setting The Current Group And Synchronising Merge And Diff View Position).

The Button bar also contains nine navigation buttons with which you can jump to the current/-first/last difference, to the previous/next difference (**Ctrl-Down**/**Ctrl-Up**), to the previous/next conflict (**Ctrl-PgDn**/**Ctrl-PgUp**), or to the previous/next unsolved conflict. Note that for KDiff3 a "conflict" that was not automatically solved at the start of the merge stays a "conflict" even if it is solved. Hence the necessity to distinguish "unsolved conflicts".

2.6.1 Auto-Advance

There also is a button **Automatically Go to Next Unsolved Conflict After Source Selection** (Auto-Advance). If you enable this, then, when one source is selected, KDiff3 will jump to and select the next unsolved conflict automatically. This can help when you always want to choose one source only. When you need both sources, or you want to edit after selecting, then you probably want to switch this off. Before proceeding to the next unsolved conflict KDiff3 shows you the effect of your choice for a short time. This delay is adjustable in the **Merge** item in the settings dialog: You can specify the **Auto advance delay (ms):** in milli seconds between 0 and 2000. Hint: Tired of many clicks? - Use a small Auto-Advance-delay and the shortcuts **Ctrl-1/2/3** to select **A/B/C** for many conflicts.

2.7 Select, Copy And Paste

The input windows don't show a cursor, so selections must be made with the mouse by clicking with the right mouse button at the start, holding down the mousebutton and moving to the end, where you release the mouse button again. You can also select a word by double clicking it. In the merge output editor you can also select via the keyboard by holding the **Shift** key and navigation with the cursor keys.

If the selection exceeds the visible range you can move the mouse over the window borders which causes KDiff3 to scroll in that direction.

For very large selections you can also use the navigation keys while holding down the mouse. E.g. use **PgUp** and **PgDn** to quickly go to a certain position. At the end position release the mouse button.

In order to select everything in the current window use **Edit** → **Select All** menu item (**Ctrl-A**).

To copy to the clipboard you must press the **Copy** button (**Ctrl-C** or **Ctrl-Ins**). But there exists an option **Auto copy selection** at **Editor** item in the settings dialog. If this is enabled, then whatever you select is copied immediately and you don't need to explicitly copy. But pay attention when using this because the contents of the clipboard might then be destroyed accidentally.

Cut (**Ctrl-X** or **Shift-Del**) copies to the clipboard and deletes the selected text.

Paste (**Ctrl-V** or **Shift-Ins**) inserts the text in the clipboard at the cursorposition or over the current selection. If you paste to either diff input window the contents of the clipboard will be shown in that window and the comparison will restart immediately. This is useful if you want to quickly grab a piece of text from somewhere and compare it with something else without first creating files.

2.8 Saving

Saving will only be allowed, when all conflicts were solved. If the file already exists and the **Backup files** option is enabled at **Folder** item in the settings dialog then the existing file will be renamed with an `.orig` extension, but if such a file exists it will be deleted. When you exit or start another diff-analysis and data wasn't saved yet, then KDiff3 will ask if you want to save, cancel or proceed without saving. (KDiff3 does not catch any signals. So if you "kill" a KDiff3 instance then your data will be lost.)

Line endings are saved according to the normal method on the underlying operating system. For Unices each line ends with an linefeed-character `"\n"`, while for Win32-based systems each line ends with a carriage-return + a linefeed `"\r\n"`. KDiff3 does not preserve the line-endings of the input files, which also means that you shouldn't use KDiff3 with binary files.

2.9 Finding Strings

You can search for a string in any text-window of KDiff3. The **Edit** → **Find...** menu item (**Ctrl-F**) opens a dialog that lets you specify the string to search for. You can also select the windows which should be searched. Searching will always start at the top. Use the **Edit** → **Find Next...** menu item (**F3**) to proceed to the next occurrence. If you select to search several windows then the first window will be searched from top to bottom before the search starts in the next window at the top again, etc.

2.10 Printing

KDiff3 supports printing for textfile differences. The **File** → **Print...** menu item (**Ctrl-P**) opens a dialog that allows you to select the printer and to adjust other options.

There are several possibilities to adjust the range. Due to different printing dialogs on different operating systems, the method to achieve certain range selections varies.

All:
Print everything.

Current:
Print a page starting at the first visible line in the window. (On systems without this option this can be achieved by specifying page number 10000 for printing.)

Selection:
Before choosing to print select text with the mouse (like for copy and paste) in one of the diff input windows to define the start and end line. If no text in one of the diff input windows was selected, then this won't be an available choice. (On systems without this option this can be achived by specifying page number 9999 for printing.)

Range:
Specify the first and last page.

Other important options for printing will be taken from the normal options:

- Font, font size
- Show line numbers
- Word wrap
- Colors
- etc.

Landscape formatting is also recommended for printing.

2.11 Options

Options and the recent-file-list will be saved when you exit the program, and reloaded when you start it. (**Settings** → **Configure KDiff3...** menu item).

2.11.1 Font

Select a fixed width font. (On some systems this dialog will also present variable width fonts, but you should not use them.)

2.11.2 Colors

Editor and Diff Views:

Foreground color:
Usually black.

Background color:
Usually white.

Diff background color:
Usually light gray.

Color A:
Usually dark blue.

Color B:
Usually dark green.

Color C:
Usually dark magenta.

Conflict color:
Usually red.

Current range background color:
Usually light yellow.

Current range diff background color:
Usually dark yellow.

Color for manually aligned difference ranges:
Usually orange.

Folder Comparison View:

Newest file color:
Usually green.

Oldest file color:
Usually red.

Middle age file color:
Usually strong yellow.

Color for missing files:
Usually black.

Changing the colors for folder comparison will be effective only when starting the next folder comparison.

On systems with only 16 or 256 colors some colors are not available in pure form. On such systems the **Defaults** button will choose a pure color.

2.11.3 Editor Settings

Tab inserts spaces

If this is disabled and you press the **Tab** key, a tab-character is inserted, otherwise the appropriate amount of characters is inserted.

Tab size:

Can be adjusted for your specific needs. Default is `8`.

Auto indentation

When pressing **Enter** or **Return** the indentation of the previous line is used for the new line.

Auto copy selection

Every selection is immediately copied to the clipboard when active and you needn't explicitly copy it.

Line end style:

When saving you can select what line end style you prefer. The default setting is the common choice for the used operating system.

2.11.4 Diff Settings

When comparing files, KDiff3 first it tries to match lines that are equal in all input files. Only during this step it might ignore white space. The second step compares each line. In this step white space will not be ignored. Also during the merge white space will not be ignored.

Ignore numbers (treat as white space)

Default is off. Number characters ('0'-'9', '.', '-') will be ignored in the first part of the analysis in which the line matching is done. In the result the differences will be shown nevertheless, but they are treated as white space.

Ignore C/C++ comments (treat as white space)

Default is off. Changes in comments will be treated like changes in white space.

Ignore case (treat as white space)

Default is off. Case-differences of characters (like 'A' vs. 'a') will be treated like changes in white space.

Preprocessor command:

See next section.

Line-matching preprocessor command:

See next section.

Try hard (slower)

Try hard to find an even smaller delta. (Default is on.) This will probably be effective for complicated and big files. And slow for very big files.

Align B and C for 3 input files

Try to align **B** and **C** when comparing or merging three input files. Not recommended for merging because merge might get more complicated. (Default is off.)

2.11.5 Merge Settings

Auto advance delay (ms):

When in auto-advance-mode this setting specifies how long to show the result of the selection before jumping to the next unsolved conflict.

Show info dialogs

Show a dialog with information about the number of conflicts.

White space 2/3-file merge default:

Automatically solve all white-space conflict by choosing the specified file. (Default is manual choice.) Useful if white space really isn't important in many files. If you need this only occasionally better use **Choose A/B/C for All Unsolved Whitespace Conflicts** in the **Merge** menu. Note that if you enable either **Ignore numbers (treat as white space)** or **Ignore C/C++ comments (treat as white space)** then this auto-choice also applies for conflicts in numbers or comments.

Auto merge regular expression:

Regular expression for lines where KDiff3 should automatically choose one source. See also Automatic Merge....

Run regular expression auto merge on merge start

If activated KDiff3 runs the automatic merge using the **Auto merge regular expression:** option when a merge is started.

History entry start regular expression:

Regular expression for the start of the merge history entry. Usually this line contains the "Log" keyword. Default value: ".*\$Log.*\$.*"

History entry start regular expression:

A merge history entry consists of several lines. Specify the regular expression to detect the first line (without the leading comment). Use parentheses to group the keys you want to use for sorting. If left empty, then KDiff3 assumes that empty lines separate history entries. See also Automatic Merge....

History merge sorting

Enable version control history sorting.

History entry start sort key order:

Each pair of parentheses used in the regular expression for the history start entry groups a key that can be used for sorting. Specify the list of keys (that are numbered in order of occurrence starting with 1) using ',' as separator (e.g. "4,5,6,1,2,3,7"). If left empty, then no sorting will be done. See also Automatic Merge....

Merge version control history on merge start

If activated KDiff3 runs the automatic history merging using aforementioned options when a merge is started.

Max number of history entries:

KDiff3 truncates the history list after the specified number of entries. Use -1 to avoid truncation. (Default is -1).

Test your regular expressions

This button shows a dialog that allows you to improve and test the regular expressions above. Just copy the respective data from your files into the example lines. The **Match result:** option will immediately show whether the match succeeds or not. The **Sort key result:** will display the key used for history merge sorting.

Irrelevant merge command:

Specify a command of your own that should be called when KDiff3 detects that for a three file merge the file from **B** doesn't contribute any relevant data that isn't already contained in the file from **C**. The command is called with the three filenames as parameters. Data matched by the **Auto merge regular expression** or in the history isn't considered relevant.

Auto save and quit on merge without conflicts

If KDiff3 was started for a file-merge from the command line and all conflicts are solvable without user interaction then automatically save and quit. (Similar to command line option `--auto`.)

2.11.6 Folder Merge

These options are concerned with scanning the folder and handling the merge: See the Folder Comparison/Merge Docs for details.

Yet there is one option here that is also relevant for saving single files:

Backup files (.orig)

When a file is saved and an older version already exists, then the original version will be renamed with an `.orig` extension. If an old backup file with `.orig` extension already exists then this will be deleted without backup.

2.11.7 Regional and Language Options

Use the same encoding for everything

The following encoding options can be adjusted separately for each item or if this option is true, all values will take the first value.

Note: Local Encoding is "..."

Above the codec-selectors appears this note that tells you what the local encoding is. (This is not adjustable but for your information just in case you don't know your local encoding, but need to select it.)

Auto Detect Unicode

This option attempts to use the BOM or meta data from XML/HTML documents to detect Unicode encoding. Failing that it will check if a short sample from the beginning of the file can be interpreted as UTF-8. If this check passes UTF-8 will be used. Otherwise it will fall back to the user selected codec. Only UTF-8 is supported with no BOM or metadata present.

File Encoding for A/B/C:

Adjust the file encoding for input files. This has an effect on how the special characters are interpreted. Since you can adjust each codec separately you can even compare and merge files that were saved using different codecs.

File Encoding for Merge Output and Saving:

When you have edited a file, then you can adjust which encoding will be used when saving to disk.

File Encoding for Preprocessor Files:

When you define preprocessors then they might not be able to operate on your codec. (e.g.: Your files are 16 bit unicode and your preprocessor can only take 8 bit ASCII.) With this option you can define the encoding of preprocessor output.

Right To Left Language:

Some languages are written right to left. When this option is enabled, KDiff3 draws the text from right to left in the diff input windows and in the merge output window. Note that if you start KDiff3 with the command line option `--reverse` then all layouting will be done right to left too. (This is a feature provided by Qt™.)

> NOTE
> This documentation was written assuming that this option or reverse layout are disabled. So some references to ″left″ or ″right″ must be replaced by their respective counterpart if you use these options.

2.11.8 Miscellaneous

(These options and actions are available in menus or the buttonbar.)

Overview options:

These choices are only available when you compare three files. In normal mode all differences are shown in one color-coded overview-column. But sometimes you are especially interested in the differences between only two of these three files. Selecting ″A vs. B″, ″A vs. C″ or ″B vs. C″-overview will show a second overview column with the required information next to the normal overview.

Word Wrap Diff Windows

Wrap lines when their length would exceed the width of a window.

Show Window A/B/C:

Sometimes you want to use the space on the screen better for long lines. Hide the windows that are not important. (In the Windows-menu.)

Toggle Split Orientation

Switch between diff windows shown next to each other (**A** left of **B** left of **C**) or above each other (**A** above **B** above **C**). This should also help for long lines. (In the **Window** menu.)

Merge Current File

Works if you only compare two files. A single click starts the merge and uses the filename of the last input-file as the default output filename. (When this is used to restart a merge, then the output filename will be preserved.)

Show White Space

Turn this off to suppress any highlighting of white-space-only changes in the text or overview-columns. (Note that this also applies to changes in numbers or comments if the options **Ignore numbers (treat as white space)** or **Ignore C/C++ comments (treat as white space)** are active.)

Show Space && Tabulator Characters

Sometimes the visible spaces and tabs are disturbing. You can turn this off.

Show Line Numbers

You can select if line numbers should be shown for the input files.

2.11.9 Configuring Keyboard-Shortcuts

Currently only the Frameworks version supports user-configurable keyboard-shortcuts. (Menu **Settings** → **Configure Shortcuts...**)

2.12 Preprocessor Commands

KDiff3 supports two preprocessor options.

Preprocessor command:

When any file is read, it will be piped through this external command. The output of this command will be visible instead of the original file. You can write your own preprocessor that fulfills your specific needs. Use this to cut away disturbing parts of the file, or to automatically correct the indentation etc.

Line-matching preprocessor command:

When any file is read, it will be piped through this external command. If a preprocessor-command (see above) is also specified, then the output of the preprocessor is the input of the line-matching preprocessor. The output will only be used during the line matching phase of the analysis. You can write your own preprocessor that fulfills your specific needs. Each input line must have a corresponding output line.

The idea is to allow the user greater flexibility while configuring the diff-result. But this requires an external program, and many users don't want to write one themselves. The good news is that very often **sed** or **perl** will do the job.

Example: Simple testcase: Consider file a.txt (6 lines):

```
aa
ba
ca
da
ea
fa
```

And file b.txt (3 lines):

```
cg
dg
eg
```

Without a preprocessor the following lines would be placed next to each other:

```
aa - cg
ba - dg
ca - eg
da
ea
fa
```

This is probably not wanted since the first letter contains the actually interesting information. To help the matching algorithm to ignore the second letter we can use a line matching preprocessor command, that replaces 'g' with 'a':

```
sed 's/g/a/'
```

With this command the result of the comparison would be:

```
aa
ba
ca - cg
da - dg
ea - eg
fa
```

Internally the matching algorithm sees the files after running the line matching preprocessor, but on the screen the file is unchanged. (The normal preprocessor would change the data also on the screen.)

2.12.1 sed Basics

This section only introduces some very basic features of **sed**. For more information see info:/sed or https://www.gnu.org/software/sed/manual/html_mono/sed.html. A precompiled version for Windows® can be found at http://unxutils.sourceforge.net. Note that the following examples assume that the **sed** command is in some folder in the `PATH` environment variable. If this is not the case, you have to specify the full absolute path for the command.

In this context only the **sed** substitute command is used:

```
sed 's/REGEXP/REPLACEMENT/FLAGS'
```

Before you use a new command within KDiff3, you should first test it in a console. Here the **echo** command is useful. Example:

```
echo abrakadabra | sed 's/a/o/'
-> obrakadabra
```

This example shows a very simple sed-command that replaces the first occurrence of "a" with "o". If you want to replace all occurrences then you need the "g" flag:

```
echo abrakadabra | sed 's/a/o/g'
-> obrokodobro
```

The "|"-symbol is the pipe-command that transfers the output of the previous command to the input of the following command. If you want to test with a longer file then you can use **cat** on UNIX® like systems or **type** on Windows® like systems. **sed** will do the substitution for each line.

```
cat filename | sed options
```

2.12.2 Examples For sed Use In KDiff3

2.12.2.1 Ignoring Other Types Of Comments

Currently KDiff3 understands only C/C++ comments. Using the **Line-matching preprocessor command:** option you can also ignore other types of comments, by converting them into C/C++-comments.

Example: To ignore comments starting with "#", you would like to convert them to "/ / ". Note that you also must enable the **Ignore C/C++ comments (treat as white space)** option to get an effect. An appropriate **Line-matching preprocessor command:** would be:

```
sed 's/#/\/\//'
```

Since for **sed** the "/ " character has a special meaning, it is necessary to place the "\" character before each "/ " in the replacement-string. Sometimes the "\" is required to add or remove a special meaning of certain characters. The single quotation marks (') are only important when testing on the command shell as it will otherwise attempt to process some characters. KDiff3 does not do this except for the escape sequences '\"' and '\\'.

2.12.2.2 Caseinsensitive Diff

Use the following **Line-matching preprocessor command:** to convert all input to uppercase:

```
sed 's/\(.*\)/\U\1/'
```

Here the ".*" is a regular expression that matches any string and in this context matches all characters in the line. The "\1" in the replacement string refers to the matched text within the first pair of "\(" and "\)". The "\U" converts the inserted text to uppercase.

2.12.2.3 Ignoring Version Control Keywords

CVS and other version control systems use several keywords to insert automatically generated strings (info:/cvs/Keyword substitution). All of them follow the pattern *"$KEYWORD generated text$"*. We now need a line-matching preprocessor command that removes only the generated text:

```
sed 's/\$\(Revision\|Author\|Log\|Header\|Date\).*\$/\$\1\$/'
```

The "\|" separates the possible keywords. You might want to modify this list according to your needs. The "\" before the "$" is necessary because otherwise the "$" matches the end of the line.

While experimenting with **sed** you might come to understand and even like these regular expressions. They are useful because there are many other programs that also support similar things.

2.12.2.4 Ignoring Numbers

Ignoring numbers actually is a built-in option. But as another example, this is how it would look as a line-matching preprocessor command.

```
sed 's/[0123456789.-]//g'
```

Any character within '[' and ']' is a match and will be replaced with nothing.

2.12.2.5 Ignoring Certain Columns

Sometimes a text is very strictly formatted, and contains columns that you always want to ignore, while there are other columns you want to preserve for analysis. In the following example the first five columns (characters) are ignored, the next ten columns are preserved, then again five columns are ignored and the rest of the line is preserved.

```
sed 's/.....\(..........\).....\(.*\)/\1\2/'
```

Each dot '.' matches any single character. The "\1" and "\2" in the replacement string refer to the matched text within the first and second pair of "\(" and "\)" denoting the text to be preserved.

2.12.2.6 Combining Several Substitutions

Sometimes you want to apply several substitutions at once. You can then use the semicolon ';' to separate these from each other. Example:

```
echo abrakadabra | sed 's/a/o/g;s/\(.*\)/\U\1/'
-> OBROKODOBRO
```

2.12.2.7 Using perl instead of sed

Instead of **sed** you might want to use something else like **perl**.

```
perl -p -e 's/REGEXP/REPLACEMENT/FLAGS'
```

But some details are different in **perl**. Note that where **sed** needed "\(" and "\)" **perl** requires the simpler "(" and ")" without preceding '\'. Example:

```
sed 's/\(.*\)/\U\1/'
perl -p -e 's/(.*)/\U\1/'
```

2.12.3 Order Of Preprocessor Execution

The data is piped through all internal and external preprocessors in the following order:

- Normal preprocessor,
- Line-matching preprocessor,
- Ignore case (treat as white space) (conversion to uppercase),
- Detection of C/C++ comments,
- Ignore numbers (treat as white space),
- Ignore white space

The data after the normal preprocessor will be preserved for display and merging. The other operations only modify the data that the line-matching-diff-algorithm sees.

In the rare cases where you use a normal preprocessor note that the line-matching-preprocessor sees the output of the normal preprocessor as input.

2.12.4 Warning

The preprocessor-commands are often very useful, but as with any option that modifies your texts or hides away certain differences automatically, you might accidentally overlook certain differences and in the worst case destroy important data.

For this reason during a merge if a normal preprocessor-command is being used KDiff3 will tell you so and ask you if it should be disabled or not. But it won't warn you if a **Line-matching preprocessor command:** option is active. The merge will not complete until all conflicts are solved. If you disabled **Diffview** → **Show White Space** menu item then the differences that were removed with the **Line-matching preprocessor command:** option will also be invisible. If the **Save** button remains disabled during a merge (because of remaining conflicts), make sure to enable **Diffview** → **Show White Space** menu item. If you don't want to merge these less important differences manually you can select **Merge** → **Choose [A|B|C] for All Unsolved Whitespace Conflicts** menu item.

Chapter 3

Folder Comparison and Merge with KDiff3

3.1 Introduction into Folder Comparison and Merge

Often programmers must modify many files in a folder to achieve their purpose. For this KDiff3 also lets you compare and merge complete folders recursively!

Even though comparing and merging folders seems to be quite obvious, there are several details that you should know about. Most important is of course the fact that now many files might be affected by each operation. If you don't have backups of your original data, then it can be very hard or even impossible to return to the original state. So before starting a merge, make sure that your data is safe, and going back is possible. If you make an archive or use some version control system is your decision, but even experienced programmers and integrators need the old sources now and then. And note that even though I (the author of KDiff3) try to do my best, I can't guarantee that there are no bugs. According to the GNU GPL there is NO WARRANTY whatsoever for this program. So be humble and always keep in mind:

> *To err is human, but to really mess things up you need a computer.*

So this is what this program can do for you: KDiff3 ...

- ... reads and compares two or three folders recursively,
- ... takes special care of symbolic links,
- ... lets you browse files on mouse double click,
- ... for each item proposes a merge operation, which you can change before starting the folder merge,
- ... lets you simulate the merge and lists the actions that would take place, without actually doing them,
- ... lets you really do the merge, and lets you interact whenever manual interaction is needed,
- ... lets you run the selected operation for all items (**F7** key) or the selected item (**F6** key),
- ... lets you continue the merge after manual interaction with **F7** key,
- ... optionally creates backups, with the `.orig` extension,
- ...

3.2 Starting Folder Comparison Or Merge

This is very similar to the single file merge and comparison. You just have to specify folders on the command line or in the file-open dialog.

3.2.1 Compare/Merge two folders:

```
kdiff3 folder1 folder2
kdiff3 folder1 folder2 -o destdir
```

If no destination folder is specified, then KDiff3 will use *folder2*.

3.2.2 Compare/Merge three folders:

```
kdiff3 folder1 folder2 folder3
kdiff3 folder1 folder2 folder3 -o destdir
```

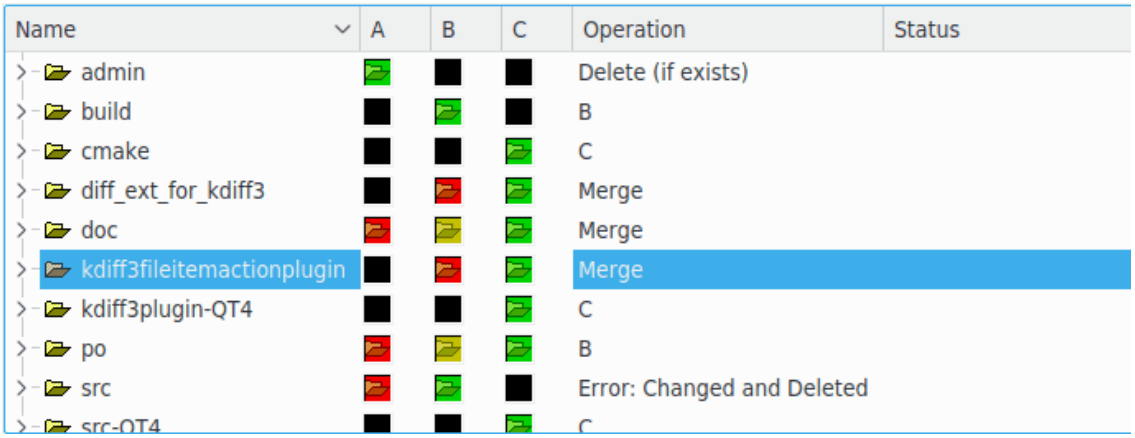
When three folders are merged then *folder1* is used as the base for the merge. If no destination folder is specified, then KDiff3 will use *folder3* as the destination folder for the merge.

Note that only the comparison starts automatically, not the merge. For this you first must select a menu entry or the **F7** key. (More details later.)

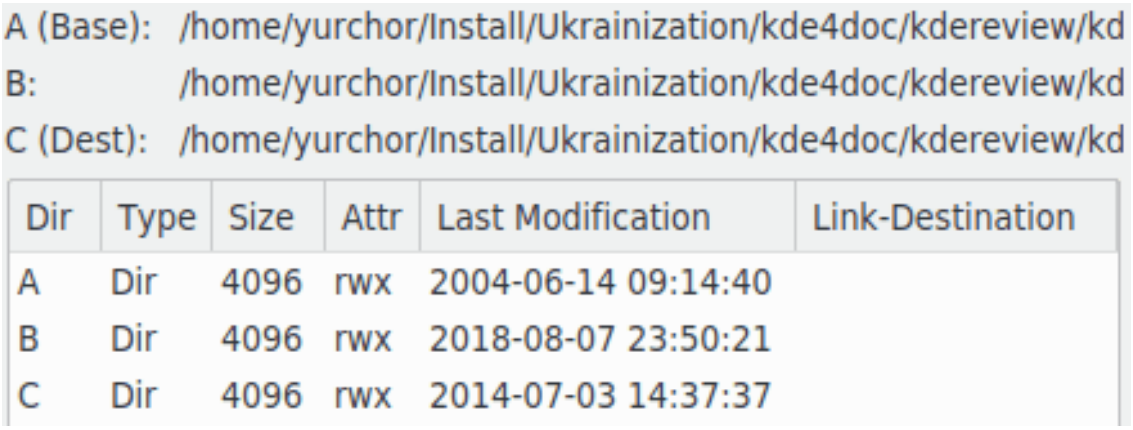
3.3 Folder Merge Visible Information

While reading the folders a message-box appears that informs you of the progress. If you abort the folder scan, then only files that have been compared until then will be listed.

When the folder scan is complete then KDiff3 will show a listbox with the results left, ...



... and details about the currently selected item on the right:



3.3.1 The Name Column

Each file and folder that was found during the scan is shown here in a tree. You can select an item by clicking it with the mouse once.

The folders are collapsed by default. You can expand and collapse them by clicking on the "+"/"-" or by double-clicking the item or by using the **Left**/**Right** cursor keys. The **Folder** menu also contains two actions **Fold All Subfolders** and **Unfold All Subfolders** with which you can collapse or expand all folders at once.

If you double-click a file item then the file comparison starts and the file-diff-window will appear.

The image in the name column reflects the file type in the first folder (**A**). It can be one of these:

- Normal file
- Normal folder (folder image)
- Link to a file (file image with a link arrow)
- Link to a folder (folder image with a link arrow)

If the file type is different in the other folders, then this is visible in the columns **A**/**B**/**C** and in the window that shows the details about the selected item. Note that for such a case no merge operation can be selected automatically. When starting the merge, then the user will be informed of problems of that kind.

3.3.2 The Columns A/B/C and the Coloring Scheme

As can be seen in the image above the colors red, green, yellow and black are used in the columns **A**/**B**/**C**.

- Black: This item doesn't exist in this folder.
- Green: Newest item.
- Yellow: Older than green, newer than red.
- Red: Oldest item.

But for items that were identical in the comparison their color also is identical even if the age is not.
Folders are considered equal if all items they contain are identical. Then they also will have the same color. But the age of a folder is not considered for its color.

The idea for this coloring scheme I came upon in dirdiff command. The colors resemble the colors of a leaf that is green when new, turns yellow later and red when old.

3.3.3 The Operation Column

After comparing the folders KDiff3 also evaluates a proposal for a merge operation. This is shown in the **Operation** column. You can modify the operation by clicking on the operation you want to change. A small menu will popup and allows you to select an operation for that item. (You can also select the most needed operations via keyboard. **Ctrl-1/2/3/4/Del** will select **A/B/C/Merge/ Delete** respectively if available.) This operation will be executed during the merge. It depends on the item and on the merge-mode you are in, what operations are available. The merge-mode is one of

- Three-folder merge (**A** is treated as older base of both).
- Two-folder merge.
- Two-folder sync-mode (activate via **Synchronize folders** option).

In three-folder merge the operation proposal will be: If for an item ...

- ... all three folders are equal: Copy from **C**
- ... **A** and **C** are equal but **B** is not: Copy from **B** (or if **B** does not exist, delete the destination if exists)
- ... **A** and **B** are equal but **C** is not: Copy from **C** (or if **C** does not exist, delete the destination if exists)
- ... **B** and **C** are equal but **A** is not: Copy from **C** (or if **C** does not exist, delete the destination if exists)
- ... only **A** exists: Delete the destination (if exists)
- ... only **B** exists: Copy from **B**
- ... only **C** exists: Copy from **C**
- ... **A**, **B** and **C** are not equal: Merge
- ... **A**, **B** and **C** don't have the same file type (e.g. **A** is a folder, **B** is a file): ″Error: Conflicting File Types″. While such items exist the folder merge cannot start.

In two-folder merge the operation proposal will be: If for an item ...

- ... both folders are equal: Copy from **B**
- ... **A** exists, but not **B**: Copy from **A**
- ... **B** exists, but not **A**: Copy from **B**
- ... **A** and **B** exist but are not equal: Merge
- ... **A** and **B** don't have the same file type (e.g. **A** is a folder, **B** is a file): ″Error: Conflicting File Types″. While such items exist the folder merge cannot start.

Sync-mode is active if only two folders and no explicit destination were specified and if the **Synchronize folders** option is active. KDiff3 then selects a default operation so that both folders are the same afterwards. If for an item ...

- ... both folders are equal: Nothing will be done.
- ... **A** exists, but not **B**: Copy **A** to **B**

- ... **B** exists, but not **A**: Copy **B** to **A**
- ... **A** and **B** exist, but are not equal: Merge and store the result in both folders. (For the user the visible save-filename is **B**, but then KDiff3 copies **B** also to **A**.)
- ... **A** and **B** don't have the same file type (e.g. **A** is a folder, **B** is a file): ″Error: Conflicting File Types″. While such items exist the folder merge cannot start.

When two folders are merged and the **Copy newer instead of merging (unsafe)** option is selected, then KDiff3 looks at the dates and proposes to choose the newer file. If the files are not equal but have equal dates, then the operation will contain ″Error: Dates are equal but files are not.″ While such items exist the folder merge cannot start.

3.3.4 The Status Column

During the merge one file after the other will be processed. The status column will show **Done** for items where the merge operation has succeeded, and other texts if something unexpected happened. When a merge is complete, then you should make a last check to see if the status for all items is agreeable.

3.3.5 Statistics Columns

When the file comparison mode **Full Analysis** is enabled in the options, then KDiff3 will show extra columns containing the numbers of Unsolved, Solved, Nonwhite and Whitespace conflicts. (The **Solved** column will only show when comparing or merging three folders.)

3.3.6 Selecting Listed Files

Several options influence which files are listed here. Some are accessible in the settings dialog. The **Folder** menu contains the entries:

- **Show Identical Files**: Files that have been detected equal in all input folders.
- **Show Different Files**: Files that exist in two or more folders but are not equal.
- **Show Files only in A**: Files that exist only in **A**, but not in **B** or **C**.
- **Show Files only in B**: Files that exist only in **B**, but not in **A** or **C**.
- **Show Files only in C**: Files that exist only in **C**, but not in **A** or **B**.

Activate only the **Show** options for the items you want listed. If for example you only want to list all items that exist either in **A** or in **B** but not in both, you'll have to activate **Show Files only in A** and **Show Files only in B** and deactivate all others (**Show Identical Files**, **Show Different Files**, **Show Files only in C**). The list will be updated immediately to reflect the change.

These options also apply for folders with one exception: Disabling **Show Different Files** will not hide any complete folders. This will work only for files within.

Note that of these only the **Show Identical Files** option is persistant. The others are enabled when starting KDiff3.

3.4 Doing A Folder Merge

You can either merge the currently selected item (file or folder), or all items. When you have made all your operation choices (in all subfolders too) then you can start the merge.

Be aware that if you didn't specify a destination folder explicitly, then the destination will be **C** in three-folder mode, **B** in two-folder merge mode, and in sync-mode it will be **A** or/and **B**.

If you have specified a destination folder also check that all items that should be in the output, are in the tree. There are some options that cause certain items to be omitted from the folder comparison and merge. Check these options to avoid unpleasant surprises:

- **Recursive Folders**: If this is off, then items in subfolders will not be found.
- **Pattern**/**Anti-Pattern**: Include/exclude items that match
- **Exclude Hidden Files**
- Show options (**Show Identical/Different Files**, **Files only in A/B/C**)

If you change the settings in order to list more files, you must do a rescan via menu **Folder** → **Rescan** yourself. (The reason for this is that for faster comparison-speed KDiff3 omits the comparison for files suppressed by these criteria.) If you changed your file and folder patterns to exclude files, then the file-list will immediately be updated on closing the options-dialog.

Note that when you write to a completely new folder then you usually also want to copy the identical files. In that case enable the **Show Identical Files** option. If your destination folder is one of the inputs, then this isn't necessary because the file is already there.

If you are satisfied so far, the rest is easy.

To merge all items: Select **Start/Continue Folder Merge** in the **Folder** menu or press **F7** (which is the default shortcut). To merge only the current item: Select **Run Operation for Current Item** or press **F6**.

If due to conflicting filetypes still some items with invalid operations exist, then a messagebox will appear and these items will be pointed out, so you can select a valid operation for the item.

If you merge all items a dialog will appear giving you the options **Do it**, **Simulate it** and **Cancel**.

- Select **Simulate it** if you want to see what would be done without actually doing it. A verbose list of all operations will be shown.
- Otherwise select **Do it** to really start merging.

Then KDiff3 will run the specified operation for all items. If manual interaction is required (single file merge), then a merge window will open (see the big screenshot).

When you have finished with manually merging a file, again select **Start/ Continue Folder Merge** or the **F7** key. If you haven't saved it yet, a dialog will ask you to do so. Then KDiff3 will continue with the next item.

When KDiff3 encounters an error, it will tell you so and will show the verbose-status-information. At the bottom of this list, there will be some error messages which should help you to understand the cause of the problem. When you continue merging (**F7** key) KDiff3 will give you the choice to retry or skip the item that caused the problem. This means that before continuing you can choose another operation or solve the problem by other means.

When the merge is complete, then KDiff3 will inform you via a message box.

If some items were merged individually before running the directorymerge then KDiff3 remembers this (while this merge-session goes on), and doesn't merge them again when later the merge for all items is run. Even when the merge was skipped or nothing was saved these items count as completed. Only when you change the merge operation the **Done** status of the item will be removed and it can be merged again.

3.5 Options for Comparing and Merging Folders

The KDiff3 preferences (menu **Settings** → **Configure KDiff3...**) has a section called ″Folder Merge″ with these options:

Recursive folders
Select whether to search folders recursively.

File pattern(s):
Only files that match any pattern here will be put in the tree. More than one pattern may be specified here by using the semicolon ″;″ as separator. Valid wildcards: ′*′ and ′?′. (e.g. ″*.cpp;*.h″). Default is ″*″. This pattern is not used on folders.

File-anti-pattern(s):
Files that match this pattern will be excluded from the tree. More than one pattern may be specified here via using the semicolon ″;″ as separator. Valid wildcards: ′*′ and ′?′. Default is ″*.orig;*.o;*.obj″.

Folder-anti-pattern(s):
Folders that match this pattern will be excluded from the tree. More than one pattern may be specified here via using the semicolon ″;″ as separator. Valid wildcards: ′*′ and ′?′. Default is ″CVS;deps;.svn″.

Use Ignore File
Ignore files and folders that would also be ignored by the your source control. Many automatically generated files are ignored using ignore lists. The big advantage is that this can be folder-specific via a local ignore file. (See version control docs for more detail.)

Find hidden files and folders
On some file systems files have an ″Hidden″-attribute. On other systems a filename starting with a dot ″.″ causes it to be hidden. This option allows you to decide whether to include these files in the tree or not. Default is on.

Follow file links
For links to files: When disabled, then the symbolic links are compared. When enabled, then the files behind the links are compared. Default is off.

Follow folder links
For links to folders: When disabled, then the symbolic links will be compared. When enabled then the link will be treated like a folder and it will be scanned recursively. (Note that the program doesn't check if the link is ″recursive″. So for example a folder that contains a link to the folder would cause an infinite loop, and after some time when the stack overflows or all memory is used up, crash the program.) Default is off.

Case sensitive filename comparison
Default is false on Windows®, true for other operating systems.

File Comparison Mode:

Binary comparison
This is the default file comparison mode.

Full analysis
Do a full analysis of each file and show the statistics information columns. (Number of **Solved**, **Unsolved**, **Nonwhite** and **White** conflicts.) The full analysis is slower than a simple binary analysis, and much slower when used on files that don't contain text. (Specify the appropriate file-antipatterns.)

Trust the size and modification date (unsafe)

If you compare big folders over a slow network, it might be faster to compare the modification dates and file length alone. But this speed improvement comes with the price of a little uncertainty. Use this option with care. Default is off.

Trust the size (unsafe)

Similar to trusting the modification date. No real comparison happens. Two files are considered equal if their file-sizes are equal. This is useful when the file-copy operation didn't preserve the modification date. Use this option with care. Default is off.

Synchronize folders

Activates sync-mode when two folders are compared and no explicit destination folder was specified. In this mode the proposed operations will be chosen so that both source folders are equal afterwards. Also the merge result will be written to both folders. Default is off.

Copy newer instead of merging (unsafe)

Instead of merging the proposed operation will copy the newer source if changes happened. (Considered unsafe, because it implies that you know, that the other file hasn't been edited too. Check to make sure in every case.) Default is off.

Backup files (.orig)

If a file or complete folder is replaced by another or is deleted then the original version will be renamed with an `.orig` extension. If an old backup file with `.orig` extension already exists then this will be deleted without backup. This also affects the normal merging of single files, not only in folder-merge mode. Default is on.

3.6 Other Functions in Folder Merge Window

3.6.1 Split/Full Screen Mode

Usually the folder-merge list view remains visible while a single file is compared or merged. With the mouse you can move the splitter bar that separates the file list from the text-diff windows. If you don't want this, you can disable **Folder** → **Folder && Text Split Screen View** menu item. Then you can use **Folder** → **Toggle View** menu item to switch between the file list and the text-diff view that then occupy the full screen.

3.6.2 Comparing or Merging a Single File

Probably you will prefer a simple double mouse click on a file in order to compare it. Nevertheless there also exists an entry in the **Folder** menu. You can also directly merge a single file by selecting it and choosing **Merge** → **Merge Current File** menu item. On saving the result, the status will be set to done, and the file will not be merged again if a folder merge is started.

But note that this status information will be lost when you rerun a folder scan: **Folder** → **Rescan**

3.6.3 Comparing or Merging Files with Different Names

Sometimes you need to compare or merge files with different names (e.g. the current file and the backup in the same folder).

Select the exact file by clicking onto the icon in the column **A**, **B** or **C**. The first file selected thus will be marked with an **A**, the second and third with **B** and **C** regardless on what column they are in. Only up to three files can be chosen like this.

Proceed by choosing **Folder** → **Compare Explicitly Selected Files** or **Folder** → **Merge Explicitly Selected Files** menu item. For your convenience these menu entries also appear as context menu when you right-click the last selected file.

The comparison or merge of a file will happen in the same window. If this method is used for folders a new window will be opened.

Chapter 4

Miscellaneous Topics

4.1 Network transparency via KIO

4.1.1 KIO-Slaves

The KIO library from Frameworks supports network transparency via KIO-slaves. KDiff3 uses this for reading input files and for scanning folders. This means that you can specify files and folders on local and remote resources via URLs.
Example:

```
kdiff3 test.cpp  ftp://ftp.faraway.org/test.cpp
kdiff3 tar:/home/hacker/archive.tar.gz/folder ./folder
```

The first line compares a local file with a file on an FTP server. The second line compares a folder within an compressed archive with a local folder.

Other KIO-slaves that are interesting are:

- Files from the WWW (http:),
- Files from the FTP (ftp:),
- Encrypted file transfer (fish:, sftp:),
- Windows® resources (smb:),
- Local files (file:),

Other things that are possible, but probably less useful are:

- Man-pages (man:),
- Info-pages (info:),

4.1.2 How To Write URLs

An URL has a different syntax compared with paths for local files and folders. Some things should be considered:

- A path can be relative and can contain ″.″ or ″..″. This is not possible for URLs which are always absolute.
- Special characters must be written with ″escaping″. (″#″ -> ″%23″, space ->″%20″, etc.) E.g. a file with the name ″#foo#″ would have the URL ″file:/%23foo%23″.
- When URLs don't work as expected, try to open them in Konqueror first.

4.1.3 Capabilities of KIO-Slaves

Network transparency has one drawback: Not all resources have the same capabilities.

Sometimes this is due to the file system of the server, sometimes due to the protocol. Here is a short list of restrictions:

- Sometimes there is no support for links.
- Or there is no way to distinguish if a link points to a file or a folder; always assuming a file. (ftp:, sftp:).
- Can't always determine the filesize.
- Limited support for permissions.
- No possibility to modify permissions or modification time, so permissions or time of a copy will differ from the original. (See the **Trust the size (unsafe)** option.) (To modify permissions or modification time is only possible for local files.)

4.2 Using KDiff3 as a KPart

KDiff3 is a KPart. Currently it implements the KParts::ReadOnlyPart interface.

It's main use is as difference-viewer in KDevelop. KDevelop always starts the internal difference viewer first. To invoke KDiff3 press the right mouse button on the difference viewer window and select **Show in KDiff3Part** from the context menu.

KDiff3 normally requires two complete files as input. When used as part KDiff3 will assume that the input file is a patch-file in the unified format. KDiff3 then retrieves the original filenames from the patch-file. At least one of the two files must be available. KDiff3 will then invoke **patch** to recreate the second file.

In Dolphin you can select a patch-file and select **Preview in** → **KDiff3Part** item from the context menu. Be aware that this won't work if none of the original files are available, and it is not reliable if the original file(s) have changed since the patch-file was generated.

When run as a part KDiff3 only provides the a two-file-diff, a very small toolbar and menu. Merging or folder comparison are not supported then.

4.3 Using KDiff3 as a Git Diff and Merging Tool

KDiff3 can be used as a Git diff and merge tool.

Just add the following lines into your `gitconfig` file.

```
[diff]
        tool = kdiff3
[difftool "kdiff3"]
        path = <path to kdiff3 binary in your system>
[difftool]
        prompt = false
        keepBackup = false
        trustExitCode = false
[merge]
        tool = kdiff3
[mergetool]
        prompt = false
```

```
        keepBackup = false
        keepTemporaries = false
[mergetool "kdiff3"]
        path = <path to kdiff3 binary in your system>
```

Then to see the difference between two commits use **`git difftool first_hash second_hash --tool=kdiff3 --cc some_file_in_the_git_tree`**

To merge a branch with KDiff3 use **`git merge branch_name && git mergetool --tool=kdiff3`**

After resolving merging conflicts in the usual way it is enough to commit the changes to do the job.

Chapter 5

Questions and Answers

This document may have been updated since your installation. You can find the latest version at https://docs.kde.org/ .

1. *Why is it called ″KDiff3″?*

   Tools named KDiff and KDiff2 (now called Kompare) already exist. Also KDiff3 should suggest that it can merge like the **diff3** tool in the Diff-Tool collection.

2. *Why did I release it under GPL?*

   I'm using GPL programs for a very long time now and learned very much by having a look at many of the sources. Hence this is my ″Thank You″ to all programmers that also did so or will do the same.

3. *Some buttons and functions are missing. What's wrong?*

   You compiled from source but you probably didn't specify the correct prefix for the installation. By default **cmake** wants to install in `/usr/local` but then the user-interface resource file (i.e. `kdiff3ui.rc`) can't be found. The `README` file contains more information about the correct prefix.

4. *Often lines that are similar but not identical appear next to each other but sometimes not. Why?*

   Lines where only the amount of white space characters is different are treated as ″equal″ at first, while just one different non-white character causes the lines to be ″different″. If similar lines appear next to each other, this actually is coincidence but this fortunately is often the case. See also Manual Diff Help.

5. *Why must all conflicts be solved before the merge result can be saved?*

   For each equal or different section the editor in the merge result window remembers where it begins or ends. This is needed so that conflicts can be solved manually by simply selecting the source button (**A**, **B** or **C**). This information is lost while saving as text and it is too much effort to create a special file format that supports saving and restoring all necessary information.

6. *How can I synchronise the diff and merge views, so that all views show the same text position?*

   Click into the summary column left of the text. (See also here.)

7. *Why does* **`git difftool --dir-diff`** *give ″Mix of links and normal files error″ when using KDiff3 as the difftool?*

   This is a side effect of git's internal workings. If you try to compare a previous revision to the current work tree git will compare actually files representing the past commit to symlinks pointing to the worktree. As of 1.9 KDiff3 defaults to having **Follow file links** and **Follow folder links** on. Prior to this they were off by default. This did not matter

when not doing if "Full analysis" was also off. However, with **Full analysis** on KDiff3 would initially attempt to compare the path pointed to by a link rather than follow it. Prior to 1.8 the resulting error was ignored. However, 1.8 began reporting the error.

8. *Why does the editor in the merge result window not have an "undo"-function?*

   This was too much effort until now. You can always restore a version from one source (**A**, **B** or **C**) by clicking the respective button. For big editing the use of another editor is recommended anyway.

9. *When I removed some text, then suddenly "<No src line>" appeared and cannot be deleted. What does that mean and how can one remove this?*

   For each equal or different section the editor in the merge result window remembers where it begins or ends. "<No src line>" means that there is nothing left in a section, not even a new line character. This can happen either while merging automatically or by editing. This is no problem, since this hint won't appear in the saved file. If you want the orignal source back just select the section (click on the left summary column) and then click the source button with the needed contents (**A**/**B** or **C**).

10. *Why doesn't KDiff3 support syntax-highlighting?*

    KDiff3 already uses many colors for difference highlighting. More highlighting would be confusing. Use another editor for this.

11. *Can I use KDiff3 to compare OpenOffice.org®, Microsoft® Word, Microsoft® Excel, PDF, etc. files?*

    Although KDiff3 will analyse any kind of file the result will probably not be very satisfactory for you.

    KDiff3 was made to compare pure text files. OpenOffice.org®, Microsoft® Word, Microsoft® Excel, etc. store much more information in the files (about fonts, pictures, pages, colors, etc.) which KDiff3 doesn't know about. So KDiff3 will show you the contents of the file interpreted as pure text, but this might be unreadable or at least it will look very odd.

    Since most programs nowadays store their contents in XML format, you might be able to read it as pure text. So if the change was only small, KDiff3 still might help you.

    The best solution if you only want to compare the text (without embedded objects like pictures) is to use **Edit** → **Select All** and **Edit** → **Copy** menu items in your program to copy the interesting text to the clipboard and then in KDiff3 paste the text into either diff input window. (See also Select, Copy And Paste.)

12. *Where has the folder option **List only deltas** gone?*

    There are now several "Show" options in the **Folder** menu. Disabling **Show Identical Files** will achieve what enabling **List only deltas** used to do.

13. *How can I make a big selection in the diff input window because scrolling takes so long?*

    Start the selection as usual (click and hold the left mouse button). Then use the navigation keys (e.g. **PgUp**, **PgDn**) while holding the left mouse button down. (See also Select, Copy And Paste.)

14. *There is so much information here, but your question is still not answered?*

    Please send me your question. I appreciate every comment.

Chapter 6

Credits and License

KDiff3 - File and Folder Comparison and Merge Tool

Several cool ideas and bugreports came from colleagues and many people out in the Wild Wild Web. Thank you!